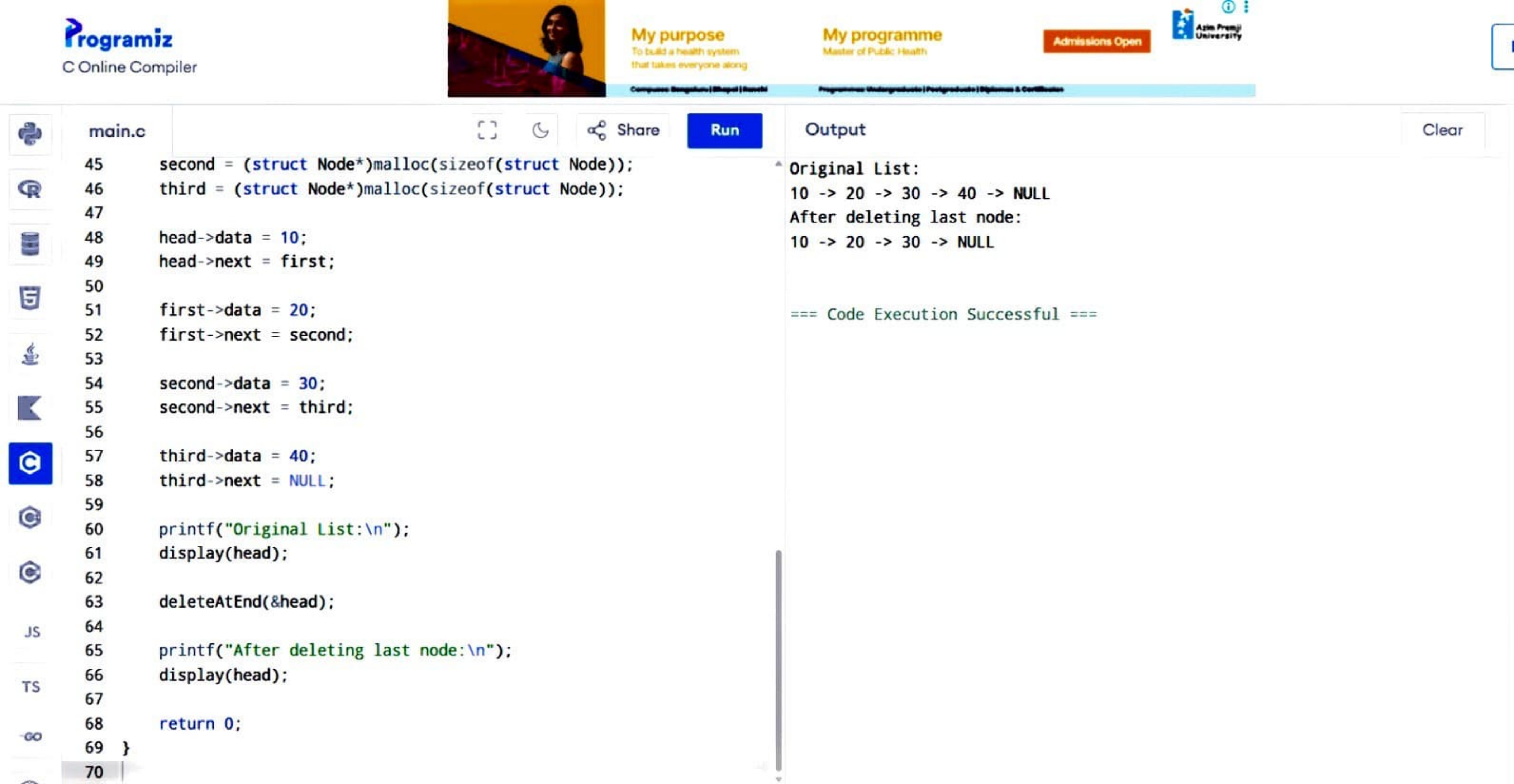
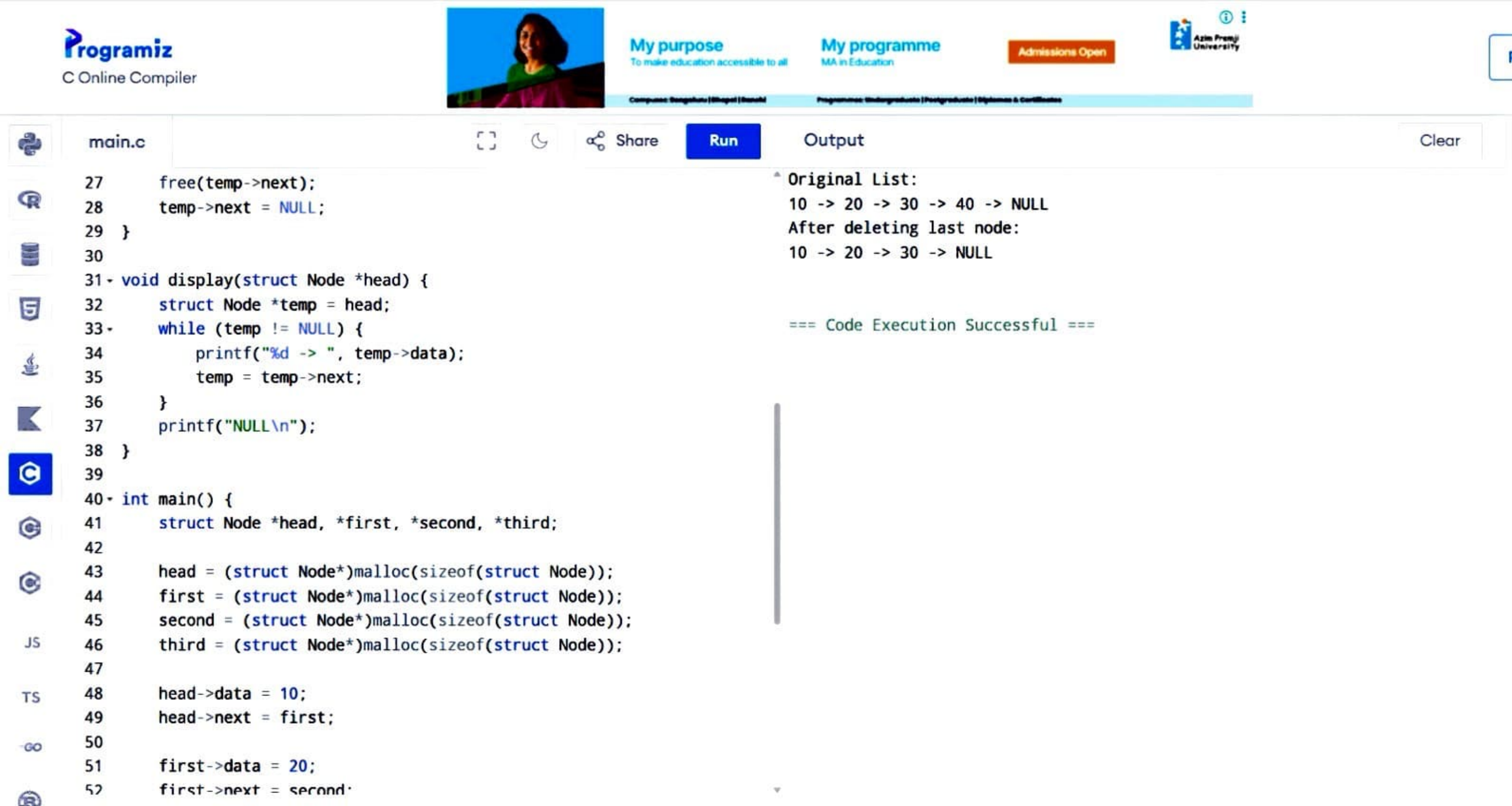


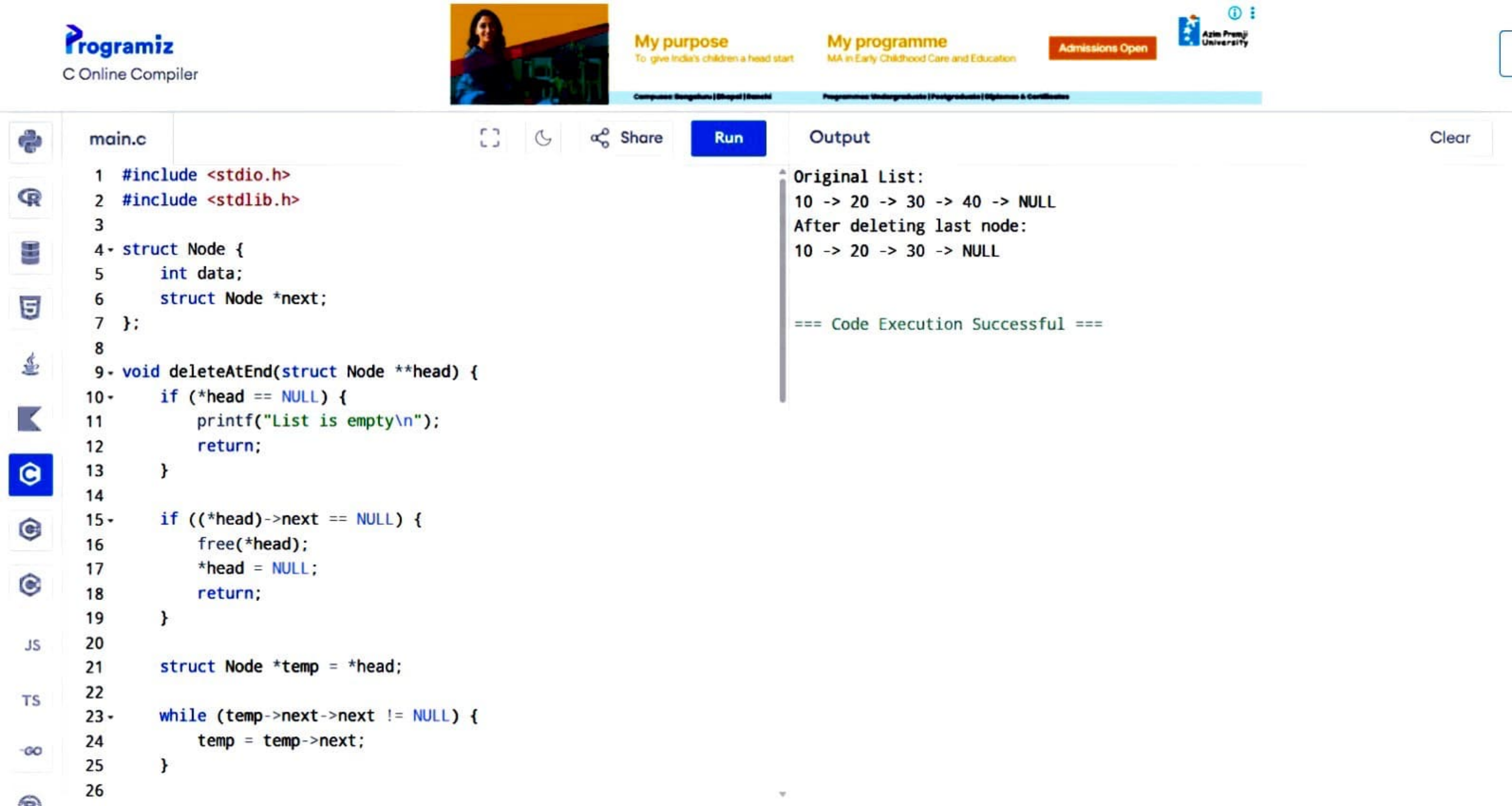
```c
    second = (struct Node*)malloc(sizeof(struct Node));
    third = (struct Node*)malloc(sizeof(struct Node));

    head->data = 10;
    head->next = first;

    first->data = 20;
    first->next = second;

    second->data = 30;
    second->next = third;

    third->data = 40;
    third->next = NULL;

    printf("Original List:\n");
    display(head);

    deleteAtEnd(&head);

    printf("After deleting last node:\n");
    display(head);

    return 0;
}
```

```
Original List:
10 -> 20 -> 30 -> 40 -> NULL
After deleting last node:
10 -> 20 -> 30 -> NULL


=== Code Execution Successful ===
```



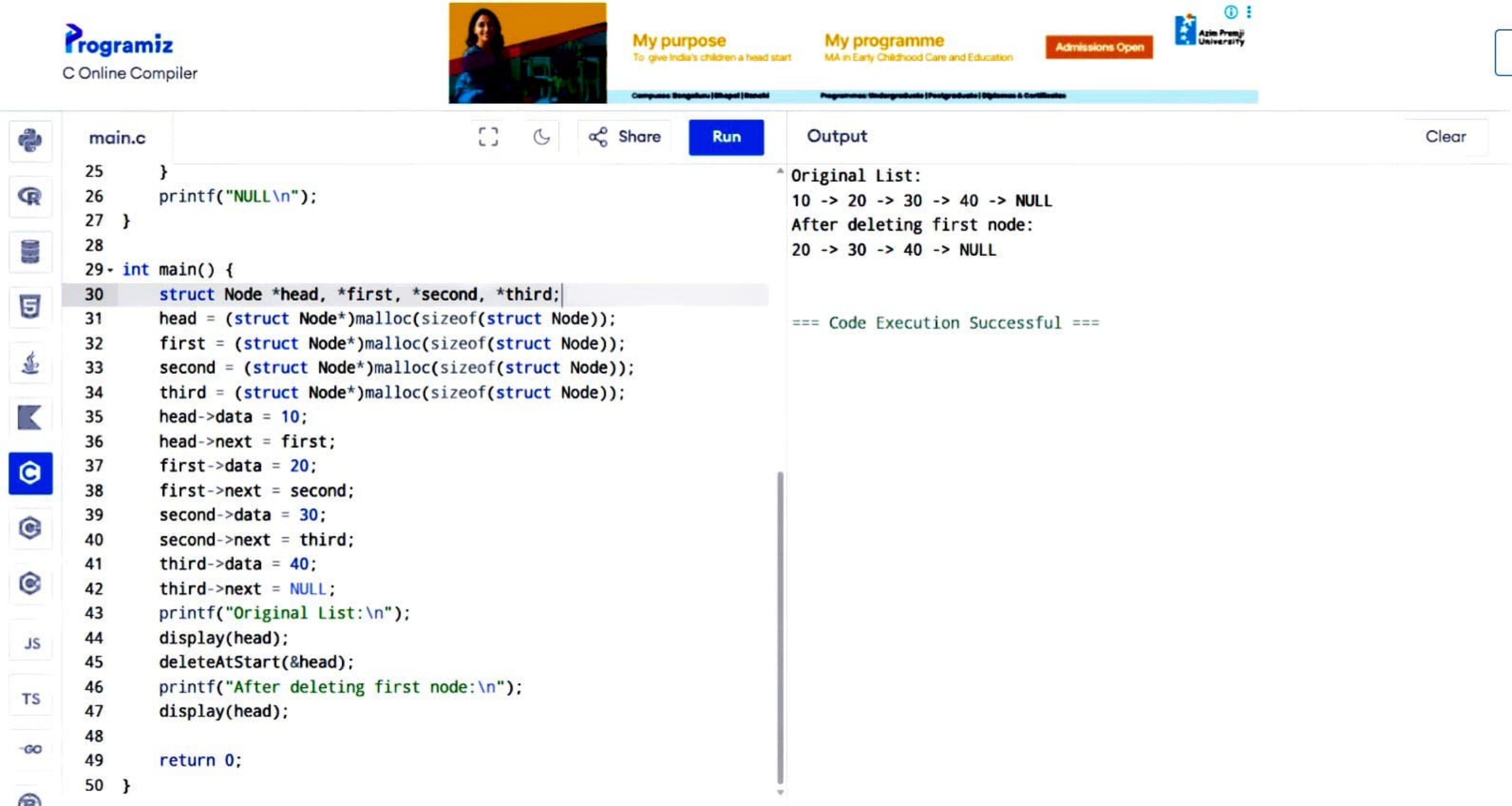
```c
    free(temp->next);
    temp->next = NULL;
}

void display(struct Node *head) {
    struct Node *temp = head;
    while (temp != NULL) {
        printf("%d -> ", temp->data);
        temp = temp->next;
    }
    printf("NULL\n");
}

int main() {
    struct Node *head, *first, *second, *third;

    head = (struct Node*)malloc(sizeof(struct Node));
    first = (struct Node*)malloc(sizeof(struct Node));
    second = (struct Node*)malloc(sizeof(struct Node));
    third = (struct Node*)malloc(sizeof(struct Node));

    head->data = 10;
    head->next = first;

    first->data = 20;
    first->next = second;
```

```
Original List:
10 -> 20 -> 30 -> 40 -> NULL
After deleting last node:
10 -> 20 -> 30 -> NULL


=== Code Execution Successful ===
```



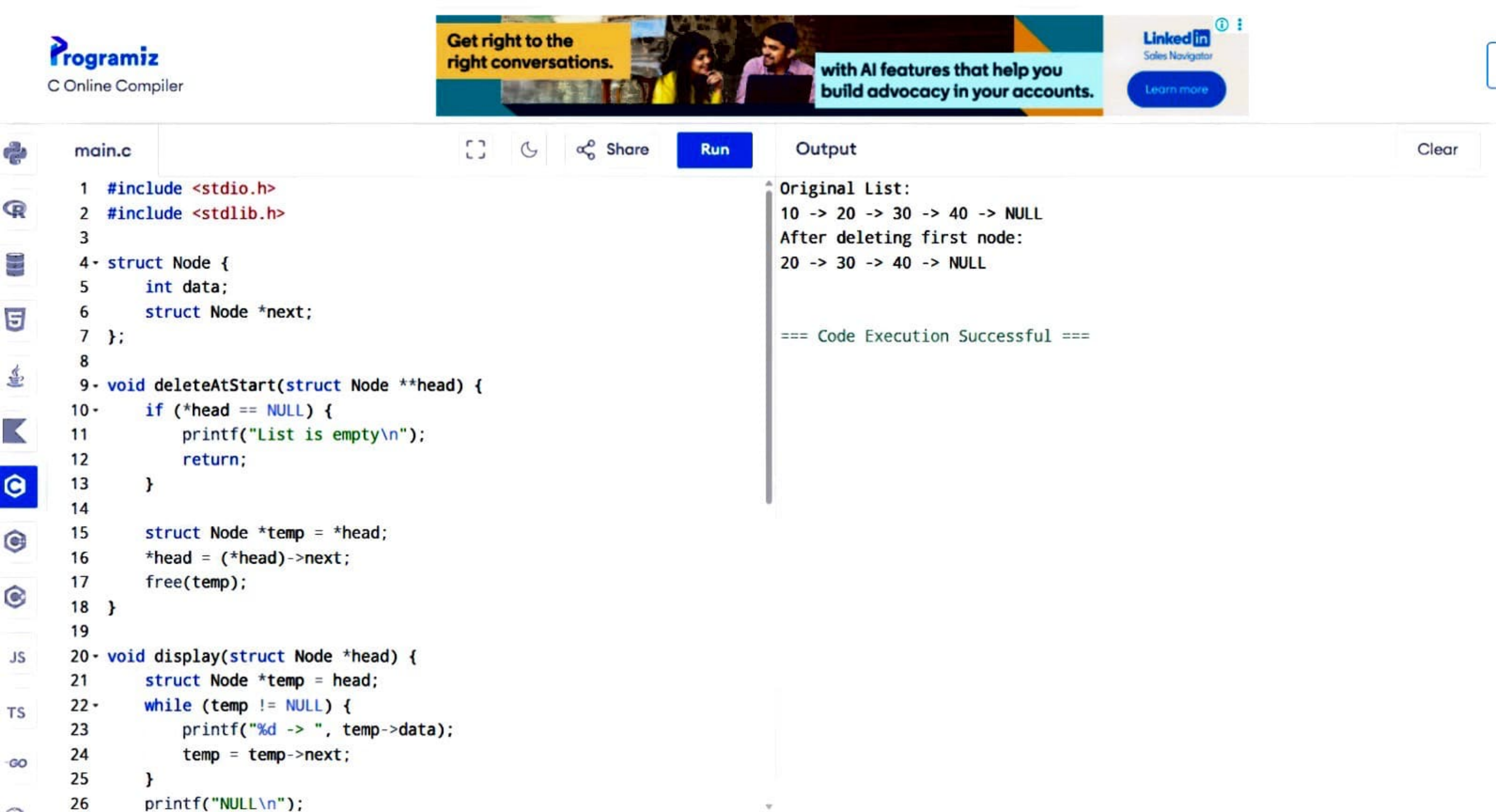
```c
#include <stdio.h>
#include <stdlib.h>

struct Node {
    int data;
    struct Node *next;
};

void deleteAtEnd(struct Node **head) {
    if (*head == NULL) {
        printf("List is empty\n");
        return;
    }

    if ((*head)->next == NULL) {
        free(*head);
        *head = NULL;
        return;
    }

    struct Node *temp = *head;

    while (temp->next->next != NULL) {
        temp = temp->next;
    }

```

```
Original List:
10 -> 20 -> 30 -> 40 -> NULL
After deleting last node:
10 -> 20 -> 30 -> NULL


=== Code Execution Successful ===
```



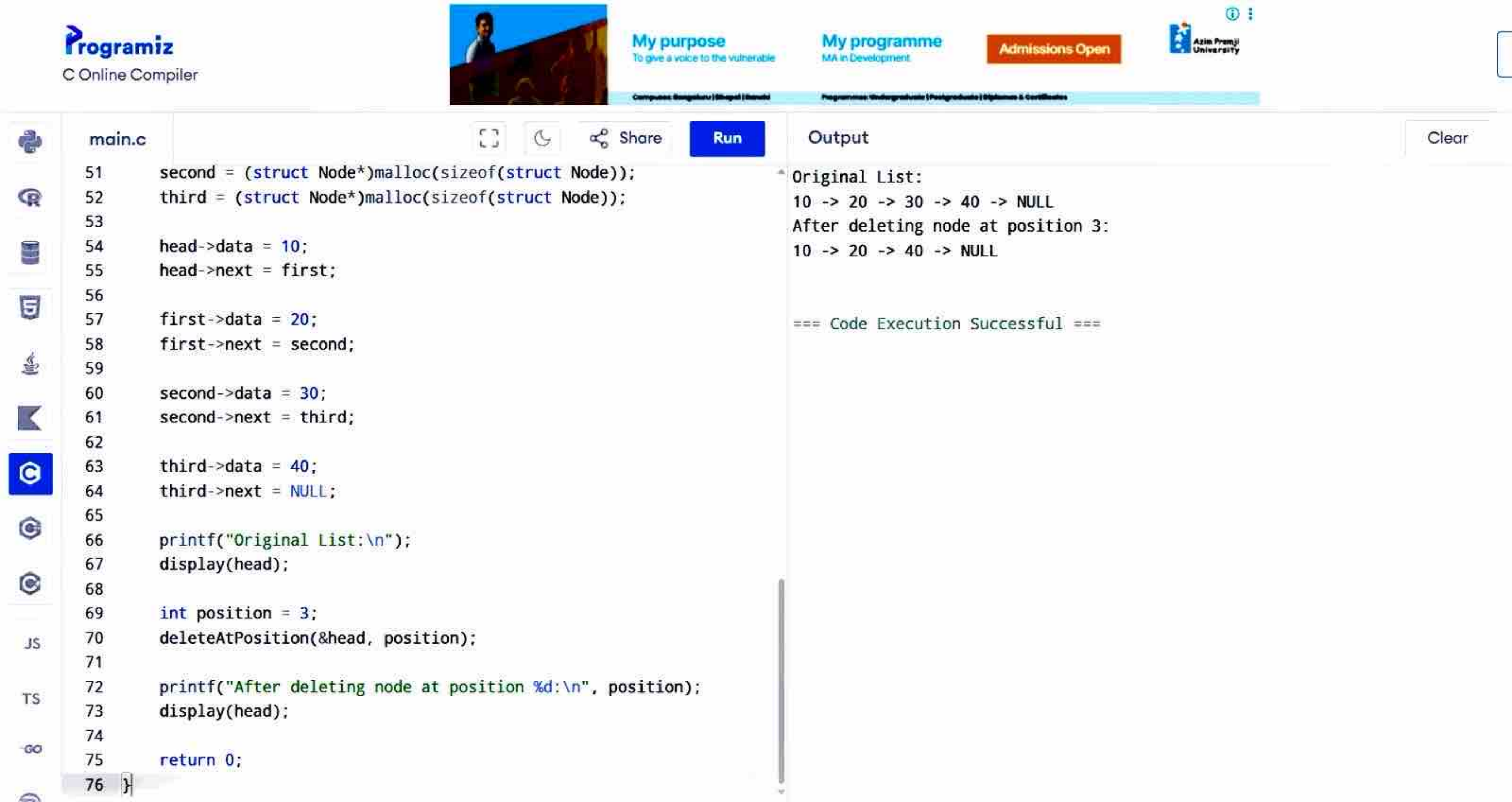
```c
    }
    printf("NULL\n");
}

int main() {
    struct Node *head, *first, *second, *third;
    head = (struct Node*)malloc(sizeof(struct Node));
    first = (struct Node*)malloc(sizeof(struct Node));
    second = (struct Node*)malloc(sizeof(struct Node));
    third = (struct Node*)malloc(sizeof(struct Node));
    head->data = 10;
    head->next = first;
    first->data = 20;
    first->next = second;
    second->data = 30;
    second->next = third;
    third->data = 40;
    third->next = NULL;
    printf("Original List:\n");
    display(head);
    deleteAtStart(&head);
    printf("After deleting first node:\n");
    display(head);

    return 0;
}
```

```
Original List:
10 -> 20 -> 30 -> 40 -> NULL
After deleting first node:
20 -> 30 -> 40 -> NULL


=== Code Execution Successful ===
```


Programiz
C Online Compiler
Get right to the right conversations.
with AI features that help you build advocacy in your accounts.
LinkedIn Sales Navigator
Learn more
main.c
Share
Run
Output
Clear

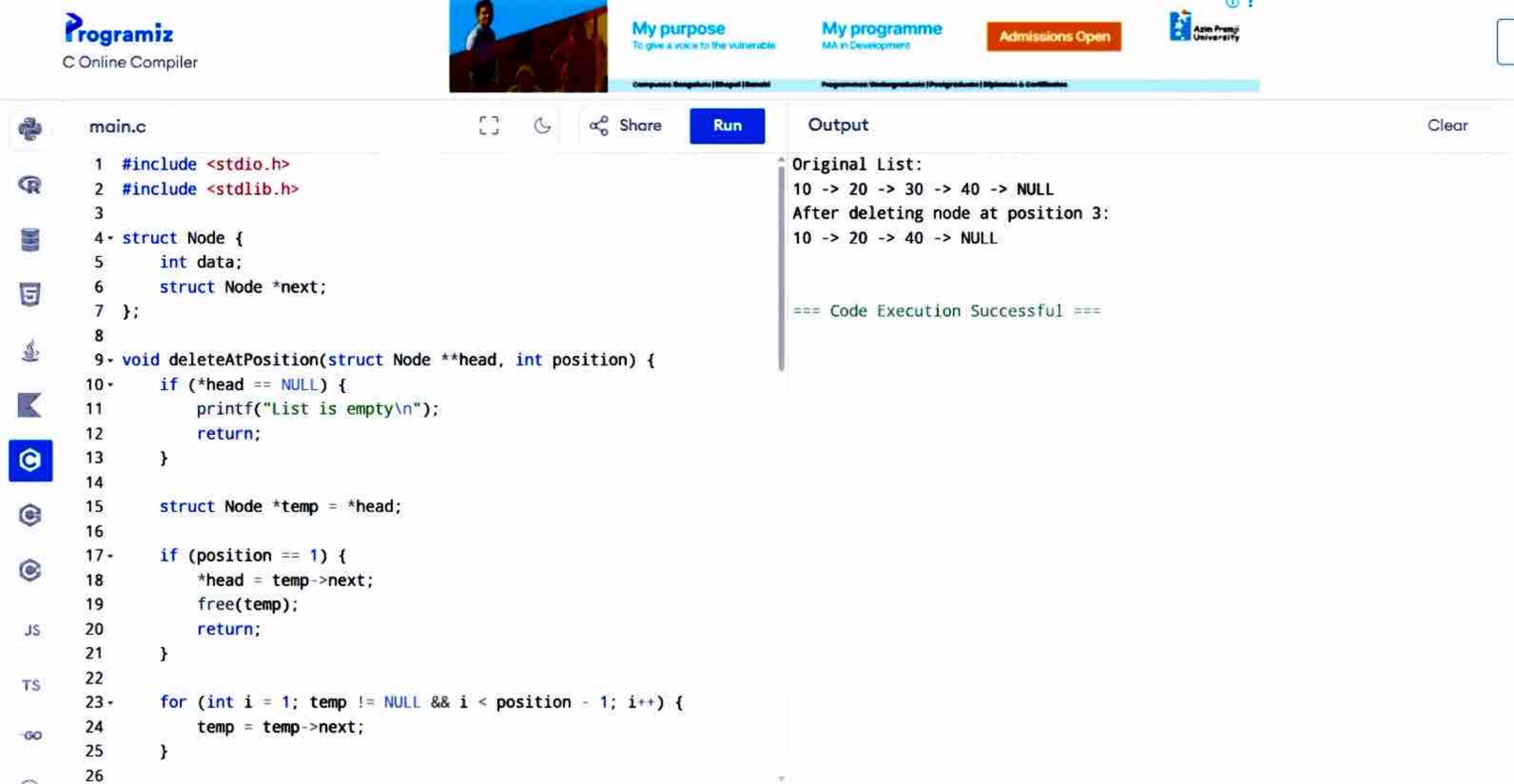
```c
#include <stdio.h>
#include <stdlib.h>

struct Node {
    int data;
    struct Node *next;
};

void deleteAtStart(struct Node **head) {
    if (*head == NULL) {
        printf("List is empty\n");
        return;
    }

    struct Node *temp = *head;
    *head = (*head)->next;
    free(temp);
}

void display(struct Node *head) {
    struct Node *temp = head;
    while (temp != NULL) {
        printf("%d -> ", temp->data);
        temp = temp->next;
    }
    printf("NULL\n");
```

```
Original List:
10 -> 20 -> 30 -> 40 -> NULL
After deleting first node:
20 -> 30 -> 40 -> NULL


=== Code Execution Successful ===
```



```c
    second = (struct Node*)malloc(sizeof(struct Node));
    third = (struct Node*)malloc(sizeof(struct Node));

    head->data = 10;
    head->next = first;

    first->data = 20;
    first->next = second;

    second->data = 30;
    second->next = third;

    third->data = 40;
    third->next = NULL;

    printf("Original List:\n");
    display(head);

    int position = 3;
    deleteAtPosition(&head, position);

    printf("After deleting node at position %d:\n", position);
    display(head);

    return 0;
}
```

```
Original List:
10 -> 20 -> 30 -> 40 -> NULL
After deleting node at position 3:
10 -> 20 -> 40 -> NULL


=== Code Execution Successful ===
```



```c
#include <stdio.h>
#include <stdlib.h>

struct Node {
    int data;
    struct Node *next;
};

void deleteAtPosition(struct Node **head, int position) {
    if (*head == NULL) {
        printf("List is empty\n");
        return;
    }

    struct Node *temp = *head;

    if (position == 1) {
        *head = temp->next;
        free(temp);
        return;
    }

    for (int i = 1; temp != NULL && i < position - 1; i++) {
        temp = temp->next;
    }

```

```
Original List:
10 -> 20 -> 30 -> 40 -> NULL
After deleting node at position 3:
10 -> 20 -> 40 -> NULL


=== Code Execution Successful ===
```